

(19)



JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: 08305563 A

(43) Date of publication of application: 22 . 11 . 96

(51) Int. Cl.

G06F 9/32

(21) Application number: 07112767

(22) Date of filing: 11 . 05 . 95

(71) Applicant: HITACHI LTD

(72) Inventor: KIUCHI ATSUSHI
HATANO YUJI

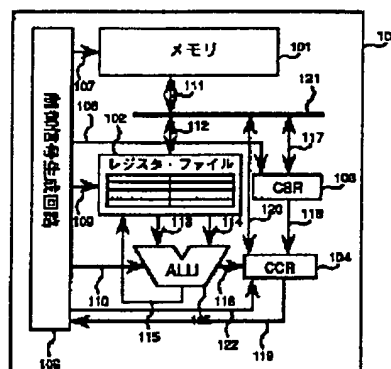
(54) DATA PROCESSING UNIT

COPYRIGHT: (C)1996,JPO

(57) Abstract

PURPOSE: To provide the data processing unit to support a conditional arithmetic instruction in which the processing efficiency is improved without causing problems such as performance deterioration due to a limit of a degree of freedom of operands or kinds of instructions and increase in program memory capacity due to increase in an instruction code length.

CONSTITUTION: The processing unit is provided with a condition selection means selecting at least one condition among plural conditions to be used for a conditional arithmetic instruction, a conditional information storage means 103 (CSR) storing condition information selected by the condition selection means, and a means 104 (CCR) storing information denoting property of the result of arithmetic operation based on at least the selected condition and the conditional arithmetic instruction has a function to execute required arithmetic processing one of selected conditions is designated and the condition is established.



This Page Blank (uspto)

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平8-305563

(43) 公開日 平成8年(1996)11月22日

(51) Int.Cl.⁶

G 0 6 F 9/32

識別記号

3 2 0

庁内整理番号

F I

G 0 6 F 9/32

技術表示箇所

3 2 0 F

審査請求 未請求 請求項の数6 O L (全 11 頁)

(21) 出願番号 特願平7-112767

(22) 出願日 平成7年(1995)5月11日

(71) 出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72) 発明者 木内 淳

東京都小平市上水本町五丁目20番1号 株式会社日立製作所半導体事業部内

(72) 発明者 波多野 雄治

東京都小平市上水本町五丁目20番1号 株式会社日立製作所半導体事業部内

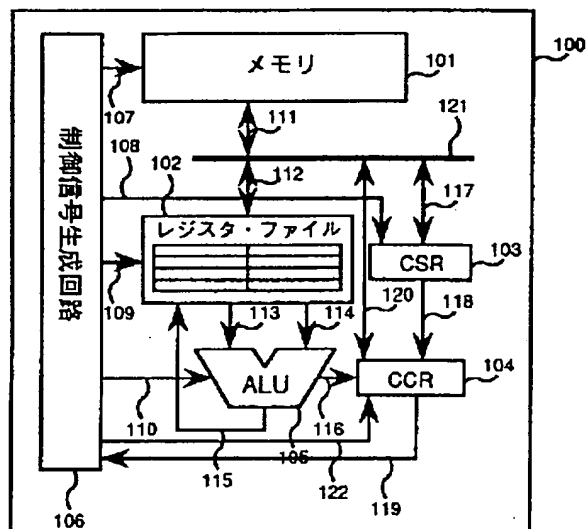
(74) 代理人 弁理士 磯村 雅俊

(54) 【発明の名称】 データ処理装置

(57) 【要約】

【目的】 命令の種類やオペランドの自由度の制限による性能低下または命令コード長の増大によるプログラムメモリ容量増加などの問題を発生することなく処理効率の向上を図ることができる条件付き演算命令をサポートするデータ処理装置を提供すること。

【構成】 条件付き演算命令に使われるべき複数の条件から、少なくとも1つの条件を選択する条件選択手段と、該条件選択手段で選択された条件情報を保持する条件情報保持手段103 (CSR) と、少なくとも前記選択された条件に基づいて演算結果の性質を表す情報を保持する手段104 (CCR) を具備し、該条件付き演算命令は該選択された条件のうちのひとつを指定して条件が成立しているときのみに、所要の演算処理を実行する機能を有する。



【特許請求の範囲】

【請求項1】 データ処理を実行する演算回路と、該データ処理に必要なソースデータを供給する手段と、演算結果を格納する演算結果格納手段と、指定された条件が成立している時のみ演算を実行する条件付き演算命令を実行する制御手段とを少なくとも具備するデータ処理装置において、該条件付き演算命令に使われるべき複数の条件から少なくとも1つの条件を選択する条件選択手段と、該条件選択手段で選択された条件情報を保持する条件情報保持手段と、少なくとも該選択された条件に基づいて演算結果の性質を表す情報を保持する手段をさらに具備し、該条件付き演算命令は該選択された条件のうちのひとつを指定して条件が成立している時のみ、所要の演算処理を実行する機能を有することを特徴とするデータ処理装置。

【請求項2】 前記演算結果の性質を表す情報を保持する保持手段は、前記選択された条件に基づいて演算結果の性質を表す情報以外に恒常的に演算結果の性質を表す情報を同時に保持する手段を持つことを特徴とする請求項1記載のデータ処理装置。

【請求項3】 前記条件選択手段は、恒常的に演算結果の性質を表す複数の情報の中からひとつを選択することを特徴とする請求項1記載のデータ処理装置。

【請求項4】 前記条件付き演算命令は、命令コード中に条件を指定するフィールドを持ち、該条件を指定するフィールドは、無条件で実行する選択肢と、前記選択された条件情報を保持する手段で保持されている条件が真の場合のみ所要の演算処理を実行する選択肢を少なくとも持つことを特徴とする請求項1記載のデータ処理装置。

【請求項5】 前記条件付き演算命令は、命令コード中に条件を指定するフィールドを持ち、該条件を指定するフィールドは、無条件で実行する選択肢と、前記選択された条件情報を保持する手段で保持されている条件が「真」の場合に所要の演算処理を実行する選択肢と、「偽」の場合に所要の演算処理を実行する選択肢とを少なくとも持つことを特徴とする請求項1記載のデータ処理装置。

【請求項6】 前記選択された条件情報を保持する手段は、前記演算結果の性質を表す情報を保持する手段と同一のレジスタで構成されていることを特徴とする請求項1記載のデータ処理装置。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明は、条件付き演算命令をサポートするデータ処理装置に関し、特に、条件付き演算命令の分岐条件指定フィールドに必要なビット数を低減することによって、命令の種類やオペランドの自由度の制限による性能低下および命令コード長の増大によるプログラムを格納するメモリ容量の増加を抑えることが可

能なデータ処理装置に関する。

【0002】

【従来の技術】 近年の高度に発展した情報処理社会においては様々な分野でデジタル信号処理が行われており、そのためのハードウェアとして、通常マイクロプロセッサやデジタルシグナルプロセッサ(DSP)と呼ばれるシステムLSIが使われている。これらのシステムLSIは、様々な用途に適用させるために、汎用の命令セットを備えているのが普通である。様々な用途のうちには、一連の固定した命令の順序に従って実行するだけでよい単純な処理もあるが、成立する条件によって実行すべき処理内容が異なってくるのが一般的である。その場合には、まず所定の条件が成立するか否かを判定し、その判定結果によって処理すべき命令を選択する必要がある。そのため従来の一般的なシステムLSIは条件分岐命令を備えている。図15に条件分岐命令による分岐処理のフローチャート例を示す。

【0003】 図15では、まず、ステップ150で所定の算術演算1を実行し、次にステップ151で前ステップ150による演算結果が正の数(0を含む)か否かを判定する。その判定結果、負の数であった場合(NO)には直接ステップ153に移り動作3を実行するが、ステップ151による判定結果が正の数(0を含む)の場合(YES)にはステップ152で動作2を実行した後、ステップ153に移る。すなわち、図15の処理例ではステップ150での実行結果によって以降の処理手順が異なっている。この処理をアセンブラ的なニモニックで表現したプログラム例を図16に示す。

【0004】 通常のシステムLSIでは、図15のような条件によって処理手順が異なる処理を実現するために、条件分岐命令を用いる。すなわち、図16に示すように1行目の算術演算命令inst.1を実行することにより、その演算結果を求める(図15のステップ150に相当)。この時、通常のシステムLSIでは同時にその演算結果の性質を表すいくつかのコードを生成する。これは通常にコンディション・コードと呼ばれており、例えば、桁上がりや桁借りが生じたことを示すフラグ(キャリー)、結果が正か負かを示すフラグ(ネガティブ)、結果がプロセッサが表現できるデータの範囲を超えたことを示すフラグ(オーバフロー)、結果がゼロかどうかを示すフラグ(ゼロ)などがその代表的なコンディション・コードのフラグ例である。

【0005】 図16の2行目では、これらのコンディション・コードのうち、演算結果が正か負かを示すフラグを用いた条件分岐命令が書かれており、もし負であれば(ifN)ラベルL1のついている行に分岐する(図15のステップ151の判定; NOに相当)。すなわち、もし負であれば3行目の命令は実行せずにスキップして4行目(図15のステップ153に相当)に分岐する。もし条件判定の結果、正(0を含む)であれば、2行目の条件

分岐命令は実行されず、3行目のinst.3を実行(図15のステップ152に相当)してから4行目に移る。このようにして図15のフローチャートに示された処理が実行できる。

【0006】ここで注目すべきことは、図15のような処理を行う場合には必ず分岐動作が必要になることである。分岐命令の役割は基本的にはプログラムの実行順序を強制的に変更することであり、データ処理には直接には何ら寄与しない動作である。しかも処理性能を向上させる極めて一般的な手法であるパイプライン制御(先行制御)を行っているシステムLSIでは、このような分岐命令によってプログラムの流れが乱されると、著しく処理効率が低下する(オーバーヘッドが増加する)。このような処理効率の低下を極力抑えるために、従来より様々な手法が提案されている。一つの手段としては、この条件分岐命令の実行によるオーバーヘッドを低減しようとするものがある。例えば条件が成立しても不成立でも同じように処理できるように、あらかじめ両方の場合で次に実行すべき命令をあらかじめプログラムメモリから読み出して用意しておく手法などが知られている。しかしこの方法はプログラムのシーケンス制御が非常に複雑になるため、ハードウェア量が增大してしまうという問題がある。

【0007】このようなハードウェアの増大を抑えて処理性能を向上させる別の手段としてADSP-2100 User's Manual p4-4~4-15に開示されているように、様々な命令に分岐命令と同様な条件を付加し、指定された条件が成立した時のみ実行し、不成立の場合は指定された動作を実行せずにそのまま次のステップに移るようにしたものが知られている。この手法を用いて図15の処理をアセンブラ的なモニクで表現したプログラム例を図17に示す。すなわち、図15のステップ152はステップ150の結果が正の時のみ実行するのであるから、図17の2行目(if P inst.2)に示されているように、1行目の演算結果として生成されるコンディション・コードのうちの正(0を含む)か負かを示すフラグが正(0を含む)を示している時のみ、inst.2を実行するような命令を備えることにより、分岐命令をプログラムから排除することができる。この2行目の命令は条件の成立/不成立に関わらず、プログラム実行のパイプラインは全く乱れないので、複雑な制御回路を増加することなしに処理性能の向上を図ることができる。

【0008】

【発明が解決しようとする課題】しかしながら、上記の条件付き演算実行命令の手法では、複雑なパイプライン制御用のハードウェアは不要になるものの、様々な命令のコードに条件指定フィールドを設けなければならないため、限られた命令コードのうちの何ビットかがそのために使われてしまい、サポートする命令全体の種類やオペランドの自由度が狭くなり、プロセッサとしての性能

が低下してしまうという問題がある。例えば、前述したような桁上がりや桁借りが生じたことを示すフラグ(キャリー)、結果が正(0を含む)か負かを示すフラグ(ネガティブ)、結果がプロセッサが表現できるデータの範囲を超えたことを示すフラグ(オーバフロー)、結果がゼロかどうかを示すフラグ(ゼロ)の4種類のコンディションの真/偽を実行の条件として指定できるようにする場合、合計8種類の条件から選択することになるので、図18に示すように命令コード90毎に少なくとも3ビットの条件指定フィールド91を確保する必要がある。3ビットの条件指定フィールド91の具体例を92に示してある。プロセッサとしての性能の低下を回避するために、この3ビットの条件指定フィールド91を従来の命令コード長に新たに追加させると、今度はプログラム全体を格納するメモリ容量が増加してしまうという別の問題が生じてしまう。

【0009】一般的にいて、上述したような条件付きで演算を実行すべきプログラムモジュールはそれほど頻繁には現れてくることはないが、例えば、デジタル信号処理の場合には一定の処理ルーチンを繰り返し実行することが多く、処理ルーチン内に1行あるだけでもそれを何回も繰り返し実行するため条件付き演算命令の処理上の性能向上の効果が大きくなる。その上、所定の処理ルーチンを繰り返し実行する際には指定されている条件はずっと同一である。本発明は、この事情を鑑みてなされたものであり、その目的とするところは、上述した如き問題点を解決し、命令の種類やオペランドの自由度の制限による性能低下、および命令コード長の増大に伴うプログラムメモリ容量の増加などの問題を発生することなく処理効率の向上を図ることができる条件付き演算命令をサポートするデータ処理装置を提供することにある。

【0010】

【課題を解決するための手段】本発明は、上記目的を達成するために、データ処理を実行する演算回路(図1の105(ALU))と、該データ処理に必要なソースデータを供給する手段(同レジスタ・ファイル102)と、演算結果を格納する演算結果格納手段(レジスタ・ファイル102)と、指定された条件が成立している時のみ演算を実行する条件付き演算命令を実行する制御手段(制御信号生成回路106)とを少なくとも具備するデータ処理装置において、該条件付き演算命令に使われるべき複数の条件から、少なくとも1つの条件を選択する条件選択手段(例えば図4の1行目)と、該条件選択手段(例えば図4の1行目)で選択された条件情報を保持する条件情報保持手段(コンディション・セレクト・レジスタ103(CSR))と、少なくとも前記選択された条件に基づいて演算結果の性質を表す情報を保持する手段(コンディション・コード・レジスタ104(CCR))をさらに具備し、該条件付き演算命令は該選択

された条件のうちのひとつを指定して条件が成立しているときのみ、所要の演算処理を実行する機能を有することを特徴としている。

【0011】また、前記演算結果の性質を表す情報を保持する保持手段（コンディション・コード・レジスタ104（CCR））は、前記選択された条件に基づいて演算結果の性質を表す情報（図2の104d（DC））以外に恒常的に演算結果の性質を表す情報（図2の104a（N）、104b（Z）、104c（V）など）を同時に保持する手段を持つことを特徴としている。また、前記条件選択手段（例えば図4の1行目）は、恒常的に演算結果の性質を表す複数の情報の中からひとつを選択することを特徴としている。

【0012】さらに、前記条件付き演算命令（図5）は、命令コード中に条件を指定するフィールド（51）を持ち、該条件を指定するフィールド（51）は、無条件で実行する選択肢（52の「0」）と、前記選択された条件情報を保持する手段で保持されている条件が真の場合のみ所要の演算処理を実行する選択肢（52の「1」）を少なくとも持つことを特徴としている。また、前記条件付き演算命令（図10）は、命令コード中に条件を指定するフィールド（81）を持ち、該条件を指定するフィールド（81）は、無条件で実行する選択肢（82の「00」）と、前記選択された条件情報を保持する手段で保持されている条件が「真」の場合に所要の演算処理を実行する選択肢（82の「01」）と、「偽」の場合に所要の演算処理を実行する選択肢（82の「10」）とを少なくとも持つことを特徴としている。また、前記選択された条件情報を保持する手段（図14の103a（CS0）、103b（CS1）、103c（CS2））は、前記演算結果の性質を表す情報を保持する手段（図14の104a（N）、104b（Z）、104c（V）と同一のレジスタ1500（CR））で構成されていることを特徴としている。

【0013】

【作用】本発明の構成において、条件選択手段（例えば図4の1行目）は選択すべき条件を規定するコード（条件情報）を条件情報保持手段（コンディション・セレクト・レジスタ103（CSR））に書き込み、以後新たに別の条件情報によって該条件情報が書き換えられるか、またはリセット処理動作などによって初期化されるまで、その条件情報を保持しておく。また、条件付き実行命令用の演算結果の性質を表す情報を保持する保持手段（図2の104d）は、コンディション・コードを更新すべき命令が実行される度に条件情報保持手段（コンディション・セレクト・レジスタ103（CSR））に保持されている情報に基づいてその内容が更新される。一方、各演算命令は、図5に示すように、無条件で実行するか条件付きで実行するかを選択するビットを持ち、条件付きで実行させる選択肢（図5の52）を持ってお

り、条件付きで演算命令が実行されると、プロセッサは該条件付き実行命令用の演算結果の性質を表す情報を保持する保持手段（コンディション・コード・レジスタ104（CCR））の内容をチェックしてその演算を実行すべきかどうかを判定する。

【0014】または、各演算命令は、図10に示すように、無条件で実行する選択肢（82の「00」）と、前記選択された条件情報を保持する手段で保持されている条件が「真」の場合に所要の演算処理を実行する選択肢（82の「01」）と、「偽」の場合に所要の演算処理を実行する選択肢（82の「10」）とを持っており、条件付きで演算命令が実行されると、プロセッサは該条件付き実行命令用の演算結果の性質を表す情報を保持する保持手段（コンディション・コード・レジスタ104（CCR））の内容をチェックして所要の演算処理を実行するか否かを判定する。この結果、各演算命令に、無条件か条件付きで実行するかを選択するビット、または、各演算命令に、無条件か、条件情報保持手段に保持されている条件が「真」の時に実行するか、「偽」の時に実行するかを選択する少数のビットを付加するだけで済み、各演算命令毎に全ての種類の条件を指定する長いフィールドを設ける必要がなくなり、命令の種類やオペランドの自由度の制限による性能低下を防止し、かつプログラムメモリの小容量化が可能になる。

【0015】

【実施例】図1は本発明の第1の実施例を説明するための構成図である。同図において、100は本発明を適用した情報処理装置の演算実行回路、101はデータを格納しているメモリ、102は演算に必要なソースデータや演算結果を格納するレジスタファイル、103は条件付き演算命令に使用するために選択されたコンディション・コード情報を保持するコンディション・セレクト・レジスタ（CSR）、104は演算結果のコンディション・コードおよび条件付き実行命令用のコンディション・コードを保持するコンディション・コード・レジスタ（CCR）、105は様々な算術演算や論理演算を実行する算術論理演算器（ALU）、106は演算実行に必要な制御信号を生成する制御信号生成回路である。

【0016】また、107は制御信号生成回路106から送られるメモリ101の読み出し、書き込みを制御する制御信号、108は制御信号生成回路106から送られるコンディション・セレクト・レジスタ103（CSR）の読み出し、書き込みを制御する制御信号、109は制御信号生成回路106から送られるレジスタ・ファイル内の各レジスタ毎の、データバス121あるいは算術論理演算器105とのデータの入出力を制御する信号群、110は制御信号生成回路106から送られる算術論理演算器105で実行する演算を選択して実行させるための制御信号群、111はメモリ101とデータバス121とを接続しているデータバス、112はレジスタ

・ファイル102とデータバス121とを接続しているデータバス、113、114はレジスタ・ファイル102から算術論理演算器105へ供給されるソース・データ、115は算術論理演算器105で実行された演算結果をレジスタ・ファイル102へ書き込むべきデスティネーション・データ、116は算術論理演算器105で実行された演算結果のコンディション・コードをコンディション・コード・レジスタ104 (CCR) に反映させるのに必要なデータ信号群である。

【0017】さらに、117はコンディション・セレクト・レジスタ103 (CSR) とデータバス121とを接続しているデータバス、118はコンディション・セレクト・レジスタ103 (CSR) に保持された選択されたコンディション・コード情報をコンディション・コード・レジスタ104 (CCR) に伝える選択情報信号、119は条件付き演算命令や通常の条件分岐命令の実行制御のために、コンディション・コード・レジスタ104 (CCR) の保持情報を制御信号生成回路106に伝えるための信号、120はコンディション・コード・レジスタ104 (CCR) とデータバス121とを接続しているデータバス、121はデータバス、122はコンディション・コード・レジスタ104 (CCR) の読み出し、書き込みを制御する信号である。なお、上記の信号は単数または複数の信号を意味し、混乱しない限りにおいて信号と信号線を同一の符号で呼ぶことがある。また本来の情報処理装置に当然含まれる他の要素回路、すなわちメモリ101に供給するアドレスおよびアドレス発生回路、命令のデコード回路やフロー制御回路などは本発明とは直接関係がないのでここでは省略している。

【0018】次に、通常の基本的な演算命令の実行手順を説明する。演算命令が実行されると、まず制御信号生成回路106は制御信号109を通じて演算に必要なソース・データをレジスタ・ファイル102から113および114に出力させる。情報処理装置が備える命令によっては、演算に必要なソース・データをメモリから供給するように指定する場合もあるが、本発明にはソース・データの出力元がどこであるかは無関係なので、以下の説明ではソース・データをレジスタ・ファイル102から供給するものと仮定する。レジスタ・ファイル102からソース・データを受け取った算術論理演算器105は、制御信号110によって指定された演算を実行し、その演算結果をデスティネーション・データ115として出力し、信号群109で指定されたレジスタ・ファイル位置 (デスティネーション・レジスタ) に入力する。この時、同時に演算結果のコンディション・コード情報をデータ信号群116を通じてコンディション・コード・レジスタ104 (CCR) に送る。

【0019】図2にコンディション・セレクト・レジスタ103 (CSR) およびコンディション・コード・レ

ジスタ104 (CCR)、およびその周辺部分の詳細な構成例を示す。図2中、103a (CS0)、103b (CS1)、103c (CS2) はコンディション・セレクト・レジスタ103 (CSR) を構成するフラグビット、200は上記各フラグビットの情報をデコードして条件付き演算命令で指定する条件をコンディション・コード・レジスタ104 (CCR) に送る選択情報信号118を生成する回路 (デコーダ)、104aは演算結果が正か負かを示すフラグ (N)、104bは演算結果がゼロかどうかを示すフラグ (Z)、104cは演算結果がプロセッサが表現できるデータの範囲を超えたことを示すフラグ (V)、104dは条件付き実行命令用のコンディション・コードを保持するフラグ (DC) である。

【0020】また、116a、116b、116c、116dは算術論理演算回路105から送られてくる演算結果のコンディション・コードをコンディション・コード・レジスタ104 (CCR) に反映させるのに必要な個別のデータ信号群、201はデータ信号群116a、116b、116c、116dおよびコンディション・セレクト・レジスタ103 (CSR) から送られてきた選択情報信号118を用いてフラグ104dに入力すべき状態信号202を生成する条件生成/選択回路である。なお、ここでは条件付き実行命令用のコンディション・コードを保持するフラグ104d以外の通常のコンディション・コードフラグとして3種類のフラグを例示しているが、この部分の種類およびその数については、特定の数に制限するものではなく、さらに不要とあれば、全くなくてもよいことは明らかである。また、状態信号202を生成するための信号116の数が、個別に反映するコンディション・コードフラグの数と異なってもよいことも明らかである。

【0021】次に、前述した図15のフローチャートの処理手順を本発明によるアセンブラ的なニモニックで表現したプログラム例を図4に示す。条件付き演算命令を実行するにあたっては、図4の1行目 (SET CSR [2:0] =101) に書かれているように、まずコンディション・セレクト・レジスタ103 (CSR) に選択すべき条件を規定する情報を設定する (条件選択手段)。ここでは、コンディション・セレクト・レジスタ103 (CSR) の第2ビット、第1ビット、第0ビットがそれぞれ1、0、1、すなわち、結果が正数 (0を含む) であることを条件としている。前述の条件付き実行命令用のコンディション・コードを保持する手段にどんなコンディションを反映させるかを予め選択する命令として、コンディション・セレクト・レジスタ103 (CSR) 内の各ビットにデータを直接セットする特別の命令を新しく設けている。具体的な動作としては、新しいセット命令からデコードされた選択情報をコンディション・セレクト・レジスタ103 (CSR) の各ビット103a、103

b, 103cに対応してコード化し、それをデータバスに出力して制御信号108をイネーブルにして書き込む方法が考えられる。

【0022】しかしコンディション・セレクト・レジスタ103(CSR)に選択すべき条件を規定する情報を設定する方法は、上述したように特に新しいセット命令を追加しなくても、レジスタ・ファイル102またはメモリ101に格納されているデータをデータ転送命令によって送り込む方法でも実現できる。またここではコンディション・セレクト・レジスタ103(CSR)を構成するフラグビットは3ビットで示しているが、ビット数は情報処理装置が条件付き演算命令に必要としている条件の種類を全て区別できればいくらかでも構わない。ここでは例として8種類の条件のうちから1つを選択するコンディション・セレクト・レジスタ103(CSR)を構成する各ビットの定義例を図3に示してある。この定義例では、結果が0でないときは000, 結果が0のときは001, 結果がオーバーフローしたときは010, 結果がオーバーフローしていないときは011, 結果が負数のときは100, 結果が正数(0を含む)のときは101, キャリーが生成されたときは110, キャリーが生成されなかったときは111としている。

【0023】コンディション・セレクト・レジスタ103(CSR)の各ビット103a(CS0), 103b(CS1), 103c(CS2)に書き込まれた情報は、以後新たに別の情報によって書き換えられるか、またはリセット処理動作などによって初期化されるまで保持され続ける。コンディション・セレクト・レジスタ103(CSR)の各ビット103a, 103b, 103cに選択すべき条件を規定する情報が設定されると、その情報をデコード200で選択されたコンディション・コード情報をコンディション・コード・レジスタ104(CCR)に伝える選択情報信号118に変換する。この変換過程は必ずしも必要という訳ではなく、103a, 103b, 103cに保持されている状態をそのまま選択情報信号118として出力しても構わない。しかしその場合は、コンディション・コード・レジスタ104(CCR)内のビット104dに格納する状態信号202を生成する全ての処理を条件生成/選択回路201で行うことになる。

【0024】コンディション・セレクト・レジスタ103(CSR)の各ビット103a, 103b, 103cに選択すべき条件を規定する情報が設定された後、コンディション・コード・レジスタ104(CCR)内のビット104dは、コンディション・コードが更新される可能性のある演算命令が実行される度にコンディション・セレクト・レジスタ103(CSR)に保持されている情報に基づいてその内容が更新される。すなわち、図4の2行目に書かれているように、コンディション・コードが更新される可能性のある演算命令(inst.1)が算

術論理演算器105で実行されると、コンディション・コード・レジスタ104(CCR)内の条件生成/選択回路201は、算術論理演算回路105から送られてきた様々な演算結果のコンディション・コード情報116と、コンディション・セレクト・レジスタ103(CSR)からの選択情報信号118によってフラグ104dに入力すべき状態信号202を生成する。

【0025】図15の処理手順では次に演算結果が正数(0を含む)であるかどうかを条件とした演算命令が必要であるので、図3に示された正数条件コード「101」が各ビット103a, 103b, 103cに保持されており、条件生成/選択回路201は演算結果のコンディション・コード情報116のうちの信号116aの反転情報を状態信号202として出力する。すなわちネガティブ信号116aが「偽」のとき、状態信号202は「真」状態を出力する。状態信号202はそのままフラグ104dに入力される。コンディション・コード・レジスタ104(CCR)内の他のフラグ104a, 104b, 104cにはそれぞれ信号116a, 116b, 116cが入力される。続いて、図4の3行目の条件付き演算命令(if P inst.2)が実行されると、前述したように2行目の演算命令の演算結果が正数(0を含む)かどうか既にフラグ104dに保持されているので、フラグ104dが「真」状態ならば指定された演算を実行し、「偽」状態ならば実行しないでそのまま次の4行目の命令(inst.3)の実行に移る。

【0026】図5に本発明における条件付き演算命令の一例を示す。同図において、50は命令コード、51は条件指定フィールド、52は条件指定フィールドの具体的なコード定義例を示している。ここでは、命令コード中に条件を指定する条件指定フィールドとして1ビット設け、該1ビットの条件指定フィールドの値によって該命令を無条件に実行するのか(0の場合)、条件付きで実行するのか(1の場合)を指定した例である。条件付きで実行するとは、選択された条件情報を保持する手段(条件情報保持手段:コンディション・セレクト・レジスタ103(CSR))に保持されている条件が真の場合にのみ所要の演算処理を実行することを意味している。

【0027】図15では、与えられた条件が「真」の時のみ追加的な処理を実行し、「偽」の時は追加処理を行わないフローを示しているが、図6に示すように「偽」の時も別の処理がある場合が考えられる。図6の処理手順をアセンブラ的なニモニックで表現したプログラム例を図7に示す。図6において、まずステップ60で算術演算1を実行し(図7の1行目)、ステップ61でその演算結果を判定し、演算結果が正数(0を含む)であれば(すなわち負数でなければ)ステップ62で動作2aを実行し(図7の3行目)、演算結果が正数(0を含む)でなければ(すなわち負数であれば)ステップ63

でステップ62とは異なる動作2bを実行し(図7の5行目)、ステップ62またはステップ63を実行した後、ステップ64において動作3を実行する(図7の6行目)。図7のプログラム例では与えられた条件の真/偽に関わらず、必ず1回は分岐命令を実行することになる。

【0028】このような例についても、本発明を適用することが可能である。本発明による図6の処理手順をアセンブラ的なニモニックで表現したプログラム例を図8に示す。図8のプログラム例によると、その2行目(if P inst. 2a) および3行目(if N inst. 2b) に示されているように分岐命令を用いず条件付き演算命令を用いている。ここで注意すべきことは、図8の2行目の条件付き演算命令(if P inst. 2a) が条件が「真」で実行された時、演算結果によってレジスタ104(CCR)のコンディション・コードを更新しないようにすることである。この制御は非常に容易であり、例えば条件付き演算命令が実行された時は制御信号122をディセーブル状態にしておくことにより、コンディション・コードの更新が行なわれないようにすることができる。このような制御は、前述した図15のような処理において同様に行なっても全く問題ないので、無条件演算実行時はコンディション・コードを更新することとし、条件付き演算実行時は条件の成立/不成立に関わらずコンディション・コードを更新しないことにすれば、図15および図6の双方の処理を本発明を用いて同一のプロセッサで実現できる。

【0029】図9は、条件付き演算命令の他の例であり、複数の条件を設定するとともに、命令中に条件を指定する条件指定フィールド71を設けたものである。この条件指定フィールドの値で指定する所定の条件が「真」のときに対応する所要の演算を行うようにしたのである。条件指定フィールド71のコード定義例としては、例えば、図9の72に示すように、無条件で実行する選択肢(72の「00」)と、前記選択された条件情報を保持する手段で保持されている条件1が「真」の場合に所要の演算処理を実行する選択肢(72の「01」)と、前記選択された条件情報を保持する手段で保持されている条件2が「真」の場合に所要の演算処理を実行する選択肢(72の「10」)と、前記選択された条件情報を保持する手段で保持されている条件3が「真」の場合に所要の演算処理を実行する選択肢(72の「11」)とを有している。

【0030】また、図10は、条件付き演算命令の別の例である。この条件付き演算命令は命令中に条件を指定する条件指定フィールド81を持っている。条件指定フィールド81のコード定義例としては、例えば、同図82に示すように、無条件で実行する選択肢(82の「00」)と、前記選択された条件情報を保持する手段で保持されている条件1が「真」の場合に所要の演算処理を

実行する選択肢(82の「01」)と、保持されている条件1が「偽」の場合に所要の演算処理を実行する選択肢(82の「10」)とを有している。

【0031】本発明の実施例は他にも様々な形式が考えられる。例えば図2の別の実施例として図11に示すように、図2の条件生成/選択回路201をセレクト1100で構成し、算術論理演算回路105から送られてくる信号群116の中の1つを該セレクト1100で選択して信号202として出力し、フラグ104dに入力させてもよい。この場合、コンディション・セレクト・レジスタ103(CSR)のコード定義例としては、図12に示すように信号群116のどれを選ぶかを指定するようにすればよい。図12の定義例では、ゼロ判定信号選択のときは00、オーバフロー判定信号選択のときは01、正負判定信号選択のときは10、キャリー発生判定信号選択のときは11である。図11の実施例では、コンディション・セレクト・レジスタ103(CSR)は2ビットのレジスタになっているが、ビット数についてはこの実施例では特に2ビットでなければならないという訳ではなく、例えばより多くのコンディションから選びたい場合にはもっとビット数を増やして構わないことは明白である。

【0032】コンディション・セレクト・レジスタ103(CSR)やコンディション・コード・レジスタ104(CCR)の構成方法についても様々な方法が考えられる。図13および図14は、両者を単一のレジスタで構成した一実施例である。図13中、1500は図1のコンディション・セレクト・レジスタ103(CSR)とコンディション・コード・レジスタ104(CCR)を合体させたコントロール・レジスタCRであり、1501はコントロール・レジスタ1500とデータバス121とを接続しているデータバス、1502はコントロール・レジスタ1500と制御信号生成回路106とを接続しているデータバスである。残りは図1および図2と同一である。この実施例は、コンディション・セレクト用のビットとコンディション・コードのビットが同一のレジスタ内でビット位置が異なる以外、具体的な動作は先述の第一の実施例と同様である。

【0033】以上の実施例では、コンディション・セレクト・レジスタ103(CSR)、コンディション・コード・レジスタ104(CCR)、コントロール・レジスタ1500(CR)はすべて通常のレジスタであることを前提として述べてきたが、例えばメモリマップ上にマッピングされたレジスタであってもよいことは明らかである。

【0034】

【発明の効果】以上述べたように、本発明によれば、通常のコンディション・コード・フラグ以外に条件付き実行命令用のコンディション・コードを保持する手段を設け、該条件付き実行命令用のコンディション・コードを

保持する手段にどんなコンディションを反映させるかを予め選択する命令およびその選択情報を保持する手段を設けることにより、命令の種類やオペランドの自由度の制限による性能低下、または命令コード長の増大によるプログラムメモリ容量増加などの問題発生することなく処理効率の向上を図ることができる条件付き演算命令をサポートすることができる。

【図面の簡単な説明】

【図1】本発明の第1の実施例である。

【図2】本発明の第1の実施例の部分的な詳細ブロック図である。

【図3】コンディション・セレクト・レジスタの各ビットの定義例である。

【図4】本発明を用いて図15の処理内容をアセンブラ的なニモニックで表現したプログラム例である。

【図5】1ビットの条件指定フィールドを持つ命令コード例である。

【図6】条件付き実行動作を含む第2のフローチャート図である。

【図7】図6の処理内容をアセンブラ的なニモニックで表現したプログラム例である。

【図8】本発明を用いて図6の処理内容をアセンブラ的なニモニックで表現したプログラム例である。

【図9】2ビットの条件指定フィールドを持つ命令コード例である。

【図10】2ビットの条件指定フィールドを持つ第2の命令コード例である。

【図11】本発明の図1の実施例の別な部分的な詳細ブロック図である。

【図12】コンディション・セレクト・レジスタの各ビットの第2の定義例である。

【図13】本発明の第2の実施例を示す図である。

【図14】本発明の第2の実施例の部分的な詳細ブロック図である。

【図15】条件付き実行動作を含むフローチャート例である。

【図16】図15の処理内容をアセンブラ的なニモニックで表現したプログラム例である。

【図17】図15の処理内容をアセンブラ的なニモニックで表現した別のプログラム例である。

【図18】3ビットの条件指定フィールドを持つ命令コード例である。

【符号の説明】

100：本発明を適用した情報処理装置の演算実行回路

101：データを格納しているメモリ

102：演算に必要なソースデータや演算結果を格納するレジスタファイル

103：条件付き演算命令に使用するために選択されたコンディション・コード情報を保持するコンディション・セレクト・レジスタ (CSR)

104：演算結果のコンディション・コードおよび条件付き実行命令用のコンディション・コードを保持するコンディション・コード・レジスタ (CCR)

105：様々な算術演算や論理演算を実行する演算器 (ALU)

106：演算実行に必要な制御信号を生成する制御信号生成回路

107：メモリ101の読み出し、書き込みを制御する信号

108：コンディション・セレクト・レジスタ103 (CSR) の読み出し、書き込みを制御する信号

109：レジスタ・ファイル内の各レジスタ毎のデータバス121あるいは算術論理演算器105とのデータの入出力を制御する信号群

110：算術論理演算器105で実行する演算を選択し、実行させるための制御信号群

111：メモリ101とデータバス121とを接続しているデータバス

112：レジスタ・ファイル102とデータバス121とを接続しているデータバス

113, 114：レジスタ・ファイル102から算術論理演算器105へ供給されるソース・データ

115：算術論理演算器105で実行された演算結果をレジスタ・ファイル102へ書き込むべきデスティネーション・データ

116：算術論理演算器105で実行された演算結果のコンディション・コードをコンディション・コード・レジスタ104 (CCR) に反映させるのに必要なデータ信号群

117：コンディション・セレクト・レジスタ103 (CSR) とデータバス121とを接続しているデータバス

118：コンディション・セレクト・レジスタ103 (CSR) に保持された選択されたコンディション・コード情報をコンディション・コード・レジスタ104 (CCR) に伝える信号

119：条件付き演算命令や通常の条件分岐命令の実行制御のために、コンディション・コード・レジスタ104 (CCR) の保持情報を制御信号生成回路106に伝えるための信号

120：コンディション・コード・レジスタ104 (CCR) とデータバス121とを接続しているデータバス

121：データバス

122：コンディション・コード・レジスタ104 (CCR) の読み出し、書き込みを制御する信号

103a, 103b, 103c：コンディション・セレクト・レジスタ103を構成するフラグビット

104a：演算結果が正か負かを示すフラグ (N)

104b：演算結果がゼロかどうかを示すフラグ (Z)

104c：演算結果がプロセッサが表現できるデータの

範囲を超えたことを示すフラグ (V)

104d: 条件付き実行命令用のコンディション・コードを保持するフラグ

116 a, 116 b, 116 c, 116 d: 算術論理演算回路105から送られてくる演算結果のコンディション・コードをコンディション・コード・レジスタ104

(CCR) に反映させるのに必要な個別のデータ信号

200:各フラグビットの情報をデコードして条件付き演算命令で指定する条件をコンディション・コード・レジスタ104(CCR)に送る選択情報信号118を生 10 成する回路

201:データ信号群116a, 116b, 116c,

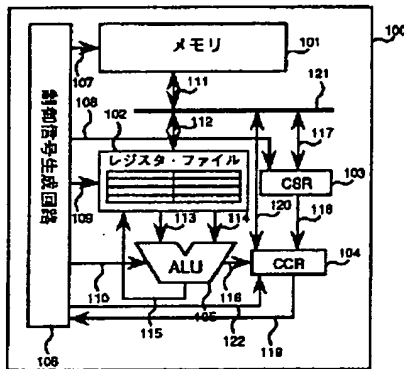
116dおよびコンディション・セレクト・レジスタ103(CSR)から送られてきた選択情報信号118を用いてフラグ104dに入力すべき状態信号202を生成する条件生成/選択回路

1500:コンディション・セレクト・ビット, 演算結果のコンディション・コードおよび条件付き実行命令用のコンディション・コードを保持するコントロール・レジスタ (CR)

1501: コントロール・レジスタ1500とデータバス121とを接続しているデータバス

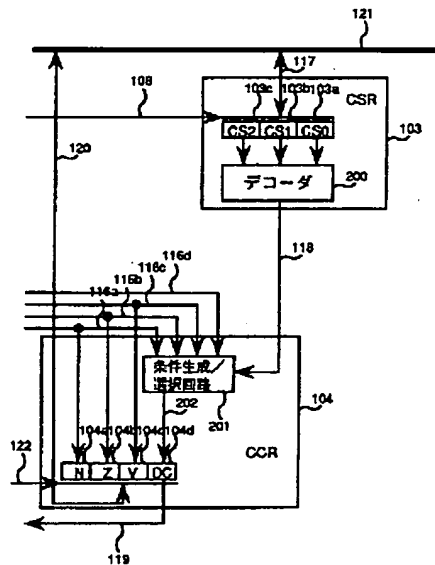
1502:コントロール・レジスタ1500の読み出し、書き込みを制御する信号

【図 1】



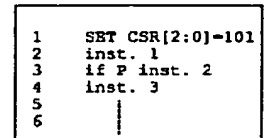
【圖 3】

【圖 2】

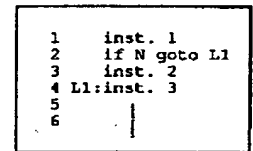


【図5】

【図 4】



【图 16】



＜コンディション、セレクト・レジスタ103のコード定義例＞

- 000:結果がゼロでない時
- 001:結果がゼロの時
- 010:結果がオーバフローした時
- 011:結果がオーバフローしていない時
- 100:結果が負数の時
- 101:結果が正数の時
- 110:キャリーが生成された時
- 111:キャリーが生成されなかった時

【図 7】

```

1      inst. 1
2      if N goto L1
3      inst. 2a
4      goto L2
5 L1:inst. 2b
6 L2:inst. 3
7      ↓
8

```

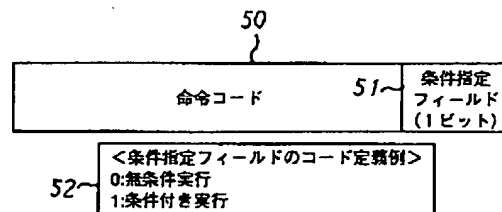
【图8】

```

1      inst. 1
2      if P inst. 2a
3      if N inst. 2b
4      inst. 3
5      |
6      |

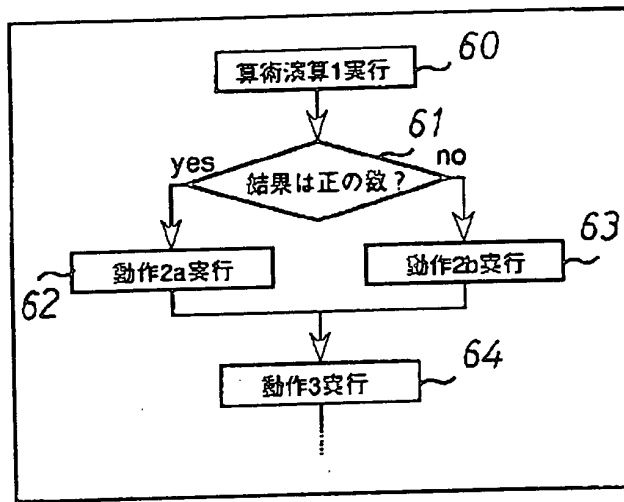
```

【図 12】

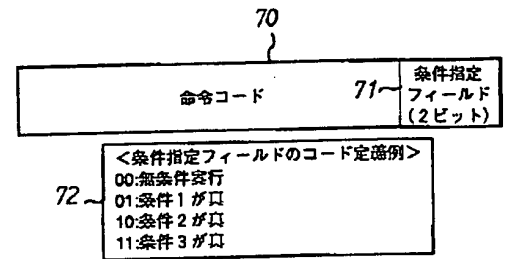


<コンディション、セレクト・レジスタ103のコード定義例>
00:ゼロ判定信号選択
01:オーバフロー判定信号選択
10:正負判定信号選択
11:キャリー発生判定信号選択

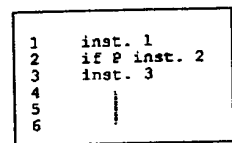
【図6】



【図9】

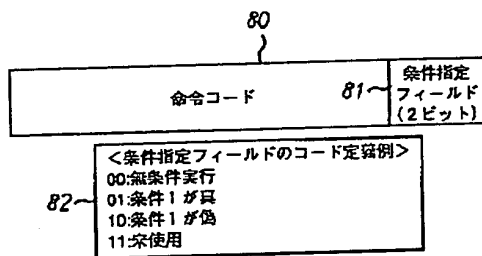


【図17】

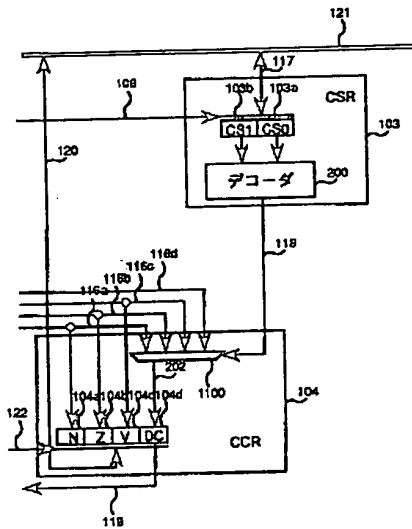
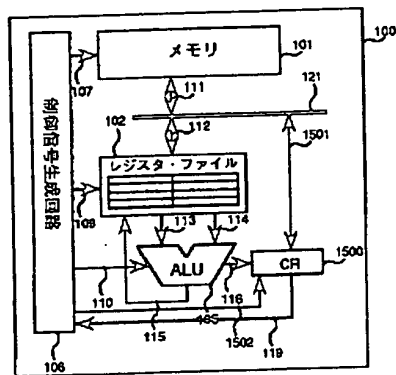


【図10】

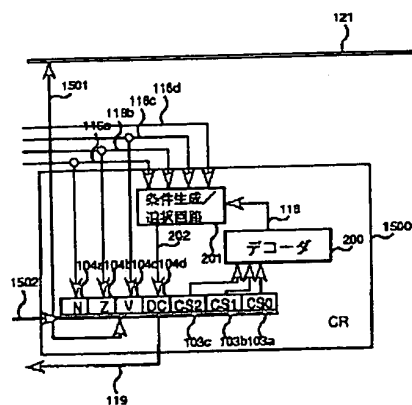
【図11】



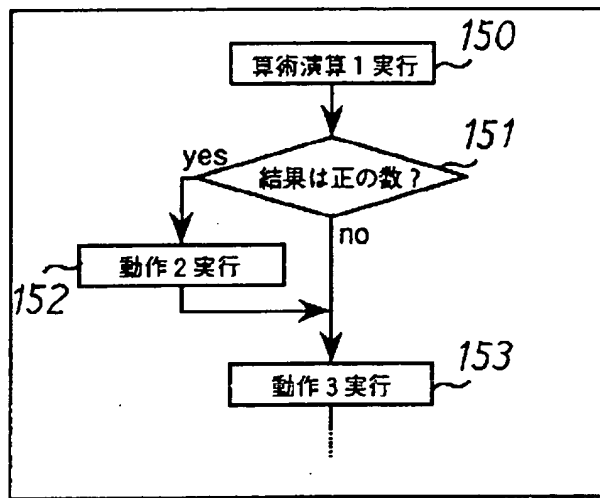
【図13】



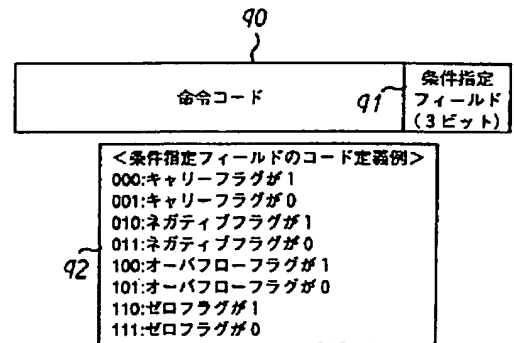
【図14】



【図15】



【図18】



This Page Blank (uspto)